REMARKS

Claims 1-53 were rejected by the Examiner. Claims 1, 15, 25, 27, 29, and 36 have been cancelled. Applicants added Claims 54-60. Claims 2-14, 16-24, 26, 28, 30-35, 37-60 are now pending. Claims 2-9, 16, 18-22, 26, 28, 30-35, 37-45, and 53 have been amended. Reconsideration is respectfully requested in view of the amendments above and the following remarks.

Claim Rejections under 35 U.S.C. § 112, 1st ¶

Claim 43 stands rejected under 35 U.S.C. § 112, 1st ¶, as failing to comply with the written description requirement. In particular, the examiner asserts that the portion of Claim 43 reciting "... the attributes required for interfacing a mobile software application with at least one of the plurality of backend applications including one or more data elements, data relationships, data dependencies, and data distribution attributes ...." Applicants respectfully traverse. However, in an effort to advance prosecution, Applicants have amended Claim 43.

In view of the amendment to Claim 43, Applicants respectfully request that the examiner reconsider the rejection of Claim 43, withdraw the rejection and allow Claim 43.

Claim Rejections under 35 U.S.C. § 103(a)

Claims 1, 3-27, 29-34, 36-38, and 53 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over U.S. Patent No. 5,857,201 issued to Wright et al. (hereinafter referred to as "Wright") and further in view of "Principles of Object Oriented Analysis and Design" by Martin (hereinafter referred to as "Martin"). Claim 2 stands rejected under 35 U.S.C. § 103(a) as being unpatentable over Wright and Martin as applied to claim 1 above, and further in view of prior art of record U.S. Patent No. 6,880,126 issued to Bahrs et al. (hereinafter referred to as "Bahrs") in view of prior art of record U.S. Patent No. 6,871,146 to Iyengar (hereinafter referred to as "Iyengar"). Claim 28 stands rejected under 35 U.S.C. § 103(a) as being unpatentable over Wright as applied to claim 27, and further in view of U.S. Patent No. 6,754,670 to Lindsay et al. (hereinafter referred to as "Lindsay"). Claim 35 stands rejected under 35 U.S.C. § 103(a) as being unpatentable over Wright as applied to claim 29 above, and further in view of U.S. Patent No. 5,604,906 issued to

Murphy et al. (hereinafter referred to as "Murphy"). Claims 39, 45-47 and 49-52 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Wright in view of Martin, and further in view of U.S. Patent No. 5,960,200 to Eager et al. (hereinafter referred to as "Eager"). Claims 40-44 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Wright in view of Martin and further in view of International Publication No. WO 00/31664 by Handel et al. (hereinafter referred to as "Handel"). Claim 48 stands rejected under 35 U.S.C. § 103(a) as being unpatentable over Wright, Martin, Eager, Bahrs, and Iyengar. Applicants respectfully traverse.

I. GOVERNING LAW

A. The legal standard for obviousness

In determining the differences between the prior art and the claims, the question under 35 U.S.C. § 103 is not whether the differences themselves would have been obvious, but whether the claimed invention as a whole would have been obvious. *Stratoflex, Inc. v. Aeroquip Corp.*, 713 F.2d 1530, 218 USPQ 871 (Fed. Cir. 1983); *Schenck v. Nortron Corp.*, 713 F.2d 782, 218 USPQ 698 (Fed. Cir. 1983) MPEP 2141.02

To establish a *prima facie* case of obviousness, three basic criteria must be met. First, there must be some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify the reference or to combine reference teachings. Second, there must be a reasonable expectation of success. Finally, the prior art reference (or references when combined) must teach or suggest all the claim limitations. MPEP 2142, citing *In re Vaeck*, 947 F.2d 488, 20 USPQ2d 1438 (Fed. Cir. 1991).

The initial burden is on the examiner to provide some suggestion of the desirability of doing what the inventor has done. "To support the conclusion that the claimed invention is directed to obvious subject matter, either the references must expressly or impliedly suggest the claimed invention or the examiner must present a convincing line of reasoning as to why the artisan would have found the claimed invention to have been obvious in light of the teachings of the references." *Ex parte Clapp*, 227 USPQ 972, 973 (Bd. Pat. App. & Inter. 1985). MPEP 2142

The teaching or suggestion to make the claimed combination and the reasonable expectation of success must both be found in the prior art, not in applicant's disclosure. *In re Vaeck*, 947 F.2d 488, 20 USPQ2d 1438 (Fed. Cir. 1991). MPEP 2143.

The examiner has not met this burden with the existing claims and the art of record will support n i such conclusion in regards to added Claims 54-60.

B. The legal standard for inherency

For a claim limitation to be inherent, it must necessarily follow that that limitation will always be present. It is well established that "[t]o establish inherency, the extrinsic evidence must make clear that the missing descriptive matter is necessarily present in the thing described in the reference, and that it would be so recognized by persons of ordinary skill. Inherency, however, may not be established by probabilities or possibilities. The mere fact that a certain thing may result from a given set of circumstances is not sufficient." *In re Robertson*, 169 F.3d 743, 745, 49 USPQ2d 1949, 1950-51 (Fed. Cir. 1999).

The fact that a certain result or characteristic may occur or be present in the prior art is not sufficient to establish the inherency of that result or characteristic. *In re Rijckaert*, 9 F.3d 1531, 1534, 28 USPQ2d 1955, 1957 (Fed.Cir.1993) (reversed rejection because inherency was based on what would result due to optimization of conditions, not what was ***necessarily*** present in the prior art); *In re Oelrich*, 666 F.2d 578, 581-82, 212 USPQ 323, 326 (CCPA 1981).

> To establish inherency, the extrinsic evidence "must make clear that the missing descriptive matter is ***necessarily*** present in the thing described in the reference, and that it would be so recognized by persons of ordinary skill. Inherency, however, may not be established by probability of possibilities. The mere fact that a certain thing ***may*** result from a given set of circumstances is not sufficient." *In re Robertson,* 169 F.3d 743, 745, 49 USPQ2d 1949, 1950-51 (Fed.Cir.1999) (citations omitted) (emphasis added).

The Patent Examiner has the burden of demonstrating that each element of Applicants' claimed invention is expressly disclosed or **necessarily** present in the prior art of record. Failure to meet that burden requires overturning the rejection. The Patent Examiner has not met this burden.

C. The claimed invention must be considered "as a whole" and "as contained in the ... claim"

"[I]n making the assessment of the differences between the prior art and the claimed subject matter, ***section 103 specifically requires consideration of the claimed invention 'as a whole.'"*** ... *See, Princeton Biochemicals, Inc. v. Beckman Coulter, Inc.*, 411 F.3d 1332 (Fed. Cir. 2002) (emphasis added). Furthermore, "the identical invention must be shown in as complete detail as is contained in the ... claim." *Richardson v. Suzuki Motor Co. Ltd.*, 868 F.2d 1226, 1236, 9 U.S.P.Q.2d 1913, 1920 (Fed. Cir. 1989).

To date, the record is replete with arguments that the examiner has not considered claimed invention as a whole, but instead made numerous rejections based on impermissible hindsight. Applicants respectfully request in consideration of Applicants' amended claims, the examiner consider the arguments presented herein, Applicants' written description and the language of amended claims "as a whole."

II. APPLICANTS' CLAIMED INVENTION: UNFROESEEN METHODOLOGY AND ADVANTAGES

As mentioned above, Claims 2-9, 16, 18-22, 26, 28, 30-35, 37-45, and 53 have been amended and Claims 54-60 have been added. Applicants respectfully assert that the amended and new claims clarify and further define that which Applicants regard as their invention. As reflected in the amended and new claims, Applicants system differs extensively from Wright and provides certain advantages Wright could not foresee because of the particulars of its implementation. Accordingly, Applicants respectfully request that the examiner withdraw the rejections to Claims 2-14, 16-24, 26, 28, 30-35, 37-53 and allow Claims 2-14, 16-24, 26, 28, 30-35, 37-60.

Specifically, Applicants' describe and claim a method and system for extending enterprise applications and data into a mobile domain. In accordance with Applicants' teachings and claims, a dedicated data model data file is created which reflects desired operating characteristics of a client-server system to extend one or more enterprise or backend application systems to one or more mobile computing devices in a mobile domain. In a particular embodiment, the dedicated data model data file describes the data elements that will be needed in the desired mobile deployment, the

connections between the data elements of the desired mobile deployment, the dependencies of the data to be utilized between the client and server of the mobile deployment, and data distribution attributes forming the policy basis for the distribution of information from the backend system to the mobile device of the mobile domain. Having guidelines for the desired operation of the mobile domain, including the interactions between the mobile computing devices and the backend system when connected, Applicants' teach the creation of at least one enterprise and mobile software application based on the guidelines defined in the dedicated data model data file. As designed and implemented, the enterprise and mobile software applications reference and access the dedicated data model data file when performing transactions, including transactions performed when the client and server are not in communication as well as during those periods of connectivity, e.g., in the performance of synchronization. However, the mobile and enterprise software applications are separate from the dedicated data model data file. In fact, the enterprise and mobile software applications are able to access the dedicated data model data file through one or more interface libraries, rather than directly. Furthermore, during periods of no connectivity between the client and server, when the client performs a transaction, the client rerecords the transaction in a mobile device transaction log along with a reference to a data model of the dedicated data model data file the client used in performing the transaction.

As described and claimed by Applicants, the dedicated data model data file is decoupled from any particular hardware or software platform, including the software platforms of the enterprise and mobile applications developed with reference to the dedicated data model data file. By providing this decoupling, changes may be made to a data model of the dedicated data model data file without requiring changes in the enterprise or mobile applications. For example, if a data model of the dedicated data model data file were originally created using data elements A and B, but circumstances had changed which now require the data model to reference data elements C, D, and E, such a change could be effected by modifying the data model of the dedicated data model data file. Once changed, at least portions of the modified dedicated data model data file would be distributed to the appropriate enterprise or mobile computing systems – such a distribution may be made in accordance with the distribution attributes of the dedicated data model data file, e.g., the distribution attributes defining which enterprise and mobile computing systems utilize the one or

more data models of the particular dedicated data model data file. Following such a change and deployment of an updated data model, the enterprise and mobile software applications may continue to perform their designed operations with reference to the updated dedicated data model data file without requiring changes to their structure.

In a world of ever-changing data, hardware and software requirements, the advantages of eliminating or minimizing the need for changes in deployed enterprise and mobile software applications are apparent. These advantages are especially clear when you compare the ease of modifications in a system such as that taught by Applicants to one which relies solely on a collection of routines, services, agents, and other programs or routines to effect its operations. An example of this latter type of system is disclosed in Wright as described below.

III. WRIGHT TEACHES A SERVER SIDE, PROGRAMMATIC SOLUTION TO CLIENT-SERVER CONNECTIVITY

A. Wright's implementation is based on a C/S architecture

In general terms, Wright discloses a FormLogic (FL) client/server system and method to access existing enterprise data sources on an occasional basis. (Abstract.) The system includes a FL builder, FL server and a FL engine. (*Id.*) The FL builder is a program to generate a communications agent that encapsulates a communication session. (*Id.*) Each session encompasses connecting the remote host, performing a specific task or set of tasks, then disconnecting from the host. (*Id.*) The FL server is connected to one or more enterprise data sources and provides the ability to link hardware devices running a FL engine as a client to access existing enterprise data sources on an occasional basis. (*Id.*) The FL engine includes a user interface, a script engine, a communications module, and a local data store. (*Id.*) Upon connection, this local database is automatically manipulated by the FL server. (*Id.*) The FL server can query the FL client database, add data to the client database, or remove data from the client database so as to make updates to both the client and server databases for reflecting changes that have happened on both sides since the last connection. (*Id.*)

i. Wright's FL Client is responsive to the FL Server

FL client 136 includes an FL Engine 160 – a hardware independent virtual machine that allows a single application to work on various hardware platforms – which allows FL applications to

execute on a variety of handheld devices. (5:16-26.) The FL Engine 160 includes a user interface (UI) 162, a script engine 164, a communications module 166, and a data store 168. (5:30-33.) The communication module 166 packages data that is either being received or sent by the FL Engine 160 and handles interfacing the FL Engine to the FL Server 132. (5:36-40.) The data store 168 includes one or more application programs 170 and a remote database 172 for storing the results of running the application program or storing data received from the FL Server 132. (5:41-44.)

The FL Engine 160 incorporates a full local database implementation that allows data to be manipulated and collected by the FL client while not connected to the FL server 132. (5:49-52.) Upon connection, this local database 172 is automatically manipulated by the FL server 132. (5:52-54.) The FL server 132 can query the client database 172, add data to the client database, or remove data from the client database in order make updates to both the client and server databases to reflect changes that have happened on both sides since the last connection. (5:54-58.) Thus, a synchronization of the two databases is performed. (5:58-59.)

The FL server 132 maintains the primary enterprise database, *e.g.*, 180. (5:60-63.) During connection, the FL server 132 is responsible for manipulation of the FL client database 172, including retrieving data that has been collected by the client since the last connection, or inserting new data in the database that has been added on the FL server 132 since the last connection. (5:63-61.) Thus, the client database 172 serves as a temporary representation of the host database. (6:1-4.) On the server side, a Remote Database API allows developers to efficiently manipulate the client database 172 with a minimum amount of data over the connection. (6:4-7.)

ii. Wright's FL Server manages and effects all operations during client-server connectivity

To allow the FL server 132 access to any data source a developer may already be working with, an API is provided between those existing data sources, *e.g.*, 180, 182, and the FL server 132. (6:10-13.) The FL server 132 provides the ability to link hardware devices running the FL Engine 160 to access existing enterprise data sources on an occasional basis. (6:22-24.) The FL Builder is a development tool used to build FL applications that can be executed on a variety of hardware

platforms. (6:34-38.) It is unique in that it allows developers to create an object called a "communications agent" or just "agent" that encapsulates a communications "session". (6:42-45.)

Because mobile clients cannot maintain a persistent connection to the FL server 132, they must "connect" for short periods of time to perform a specified operation or set of operations. (6:46-49.) Each of these connections is referred to as a "session", during which time a specified set of operations are performed between the FL client and FL server. (6:49-51.) Each "session" encompasses connecting the remote host, performing a specific task or set of tasks, and then disconnecting from the host. (6:64-66.)

Communications agents, also just known as "agents", are developed to describe the communications "session". (6:63-64.) The agent implementation is simple, and utilizes a simple software "object" to describe the agent. (7:11-12.) The developer creates a named object and provides a name, as well as other properties, which, upon connection, tell the FL server what type of session the FL client is requesting, as well as any parameters required to perform specific operations in that session. (7:12-17.)

The FL Server 132 includes a message handler 184 for interfacing the FL Server 132 to the FL Engine 160. (7:26-29.) The message handler 184 communicates with each instantiation of the FL Connection object. (7:29-31.) Each Connection object has a current task pointer for pointing to the current task. (7:35-37.) When the task is completed, the pointer is incremented to point to the next task in the session. (7:37-38.) A set of tasks are provided or handled by a service. (7:45.) A service defines the relationship between a client application and an enterprise data source. (7:45-47.)

The FL Server APIs allow developers to write services for the FL Server that will link the FL client applications to existing enterprise data sources without worrying about multi-user and concurrency issues. (7:66-8:2.) The Service APIs fall into three distinct categories: the Remote Database APIs, the Messaging APIs, and Utility APIs. (8:17-19.)

Remote database APIs are used to directly manipulate the client database during a connection. (8:21-22.) When invoking Remote Database APIs from services, corresponding events will be passed back to the services that generated the call. (8:22-24.) Messaging APIs are used to

send specific messages to FL Agents 170 on the client device. (9:16-17.) Utility APIs don't actually send any data between the server and the client. (9:53-54.) They are used to perform functions such as setting timers, writing to the system log, and controlling the client's connection dialog. (9:54-56.)

B. Wright's client database is nothing more than a simple client data store

From the discussion above taken from Wright, the role of client database 172 is clear. Specifically, Wright expressly provides client database 172 for the limited purposes of storing the results of running the application program or storing data received from the FL Server 132. (5:41-44.) In Wright's own words:

> The FL Engine 160 incorporates a full local database implementation that allows data to be manipulated and collected by the FL client while not connected to the FL server 132. Upon connection, this local database 172 is automatically manipulated by the FL server 132. The FL server 132 can query the client database 172, add data to the client database, or remove data from the client database in order make updates to both the client and server databases to reflect changes that have happened on both sides since the last connection.

(5:52-58.) No other use of client database 172 is ever mentioned in Wright. Thus, the client database 172 of Wright is a mere data store available to retain transactions observed at the client between periods of client-server connectivity.

C. Wright uses inflexible server-side programs for all transactions during periods of client-server connectivity

From the discussion above taken from Wright, it is clear the system of Wright relies exclusively on a number of server-side APIs, agents, sessions, objects, services, etc., to control everything from enabling a remote device to connect to the server to management of the exchange of data between the two entities. Specifically, in Wright's description of the operation of the FL server 132, Wright plainly states:

> The FL server 132 provides the ability to link hardware devices running the FL Engine 160 to access existing enterprise data sources on an occasional basis. (6:22-24.)
>
> Because mobile clients cannot maintain a persistent connection to the FL server 132, they must "connect" for short periods of time, referred to as a "session",

> during which time a specified set of operations are performed between the FL client and FL server. (6:46-51.) Each "session" encompasses connecting the remote host, performing a specific task or set of tasks, and then disconnecting from the host. (6:64-66.)
>
> Communications agents, or "agents", are developed to describe the communications "session". (6:63-64.) The agent implementation is simple, and utilizes a simple software "object" to describe the agent. (7:11-12.) The developer creates a named object and provides a name, as well as other properties, which, upon connection, tell the FL server what type of session the FL client is requesting, as well as any parameters required to perform specific operations in that session. (7:12-17.)
>
> The FL Server 132 includes a message handler 184 for interfacing the FL Server 132 to the FL Engine 160. (7:26-29.) The message handler 184 communicates with each instantiation of the FL Connection object. (7:29-31.) Each Connection object has a current task pointer for pointing to the current task and when the task is completed, the pointer is incremented to point to the next task in the session. (7:35-38.) A set of tasks are provided or handled by a service. (7:45.) The service defines the relationship between a client application and an enterprise data source. (7:45-47.)
>
> The FL Server APIs allow developers to write services for the FL Server that will link the FL client applications to existing enterprise data sources without worrying about multi-user and concurrency issues. (7:66-8:2.) The Service APIs fall into three distinct categories: Remote Database APIs, Messaging APIs, and Utility APIs. (8:17-19.)
>
> Remote database APIs are used to directly manipulate the client database during a connection. (8:21-22.) When invoking Remote Database APIs from services, corresponding events will be passed back to the services that generated the call. (8:22-24.) Messaging APIs are used to send specific messages to FL Agents 170 on the client device. (9:16-17.) Utility APIs don't actually send any data between the server and the client. (9:53-54.) They are used to perform functions such as setting timers, writing to the system log, and controlling the client's connection dialog. (9:54-56.)

Thus, from the plain language of Wright, all transactions between a client and the server are created, characterized, controlled and managed by one or more server-side APIs, services, sessions, agents, etc. Never does Wright describe client side control of transactions, only client session requests. Consequently, there is no need for intelligence at the client – and especially in client database 172 – beyond that which it needs to record transactions observed in between periods of connectivity and, at

best, to request updates from the server. Beyond the simple database updates managed exclusively by Wright's Remote Database APIs (RDAPI), Wright does not discuss how any of the aforementioned APIs, services, sessions, agents, etc., are reliant upon any information maintained by the client database 172 to establish communications, achieve updates, or perform other interactions. Furthermore, none of the aforementioned APIs, services, sessions, agents, etc., is described by Wright as accessing any aspect of information available from the client database 172 to interface a mobile application with at least one of a plurality of backend applications – Wright includes, maintains, and uses all of this intelligence on the server side, bound up in the many routines upon which it relies to facilitate client-server connectivity.

D. The "data models" of Wright are inseparable from their corresponding routines

As admitted by the examiner, the "data models" of Wright are inherent or implied in the routines on which Wright relies to implement its system. As suggested by the examiner, in order for Wright's RDAPI to work as described, the RDAPI must have knowledge of the data structure of the client-side and/or server-side databases. Thus, the set of instructions that define the RDAPI infer or imply a data model of the client or server database. This same line a reasoning and conclusion flow for Wright's "session module." Consequently, the data models of Wright are integral with – not decoupled from – the routine from which they may be derived or in which they are implied.

IV. PROPERLY CONSIDERED – AS A WHOLE – APPLICANTS' CLAIMS ARE PATENTABLE

As described in Applicants' written description and as claimed above, Applicants do not employ the implied, suggested or otherwise unexpressed data model of Wright. Indeed, Applicants have devised a system which, based on guidelines defining the sought after interaction between mobile clients and a backend system, leverages a platform independent description of the data, transactions, distribution, and other system characteristics, desired for implementation in their system. Specifically, Applicants have devised a data model data file which defines the data elements, data relationships, desired transactions and data distributions, to be accessed by backend and mobile applications in use in the system. The backend and mobile applications of this system are designed based on the contents of the data model data file and, in carrying out their assigned

transactions, reference the same data model data file from which the applications were derived. The structure devised by Applicants permits an administrator to alter the manner in which transactions are carried out, data is arranged, and make other desired changes simply by modifying the manner in which these characteristics are described in the data model data file.

Contrasting this with the implementation of Wright, Wright's reliance on implicit, suggested, or otherwise unexpressed data models through the usage of myriad specially developed routines, APIs, applications, etc., requires to Wright to make changes across each of its routines, APIs, applications, etc., whenever Wright wants to make a database structure change, redefine the manner in which a client-server transaction proceeds, etc.

In view of the foregoing discussion of Applicants' claimed invention, its advantages and the distinctions from Wright and the other cited art, Applicants respectfully assert that the pending claims, considered as a whole, are patentable over the cited art of record. Specifically, at a minimum, Wright, alone or in combination, fails to teach, disclose or otherwise suggest:

- a dedicated data model data file referenced by an enterprise and mobile software application, as claimed by Applicants in at least Claims 39, 40, 45, and 53-60;
- a dedicated data model data file, enterprise and mobile software applications that reference and have access to, but are separate from, a dedicated data model file, and a data store separate from but instantiated in accordance with a dedicated data model data file, as claimed by Applicants in at least Claims 39, 57, 59;
- a decoupled dedicated data model data file as claimed by Applicants in at least Claims 40, 53;
- a dedicated data mode data file accessed through one or more interface libraries during transactions performed by an enterprise of mobile software application as claimed by Applicants in at least Claims 39, 53, 58 and 60; and
- a mobile software application designed and implemented in accordance with a dedicated data model data file and operable to reference a data model described in the dedicated data model data file when documenting transactions performed at the mobile computing system as claimed by Applicants in at least Claims 58 -60.

In view of the undeniable shortcomings of Wright mentioned above, the remaining cited prior art, and the advantages of Applicants' teachings, Applicants respectfully request that the

examiner reconsider the rejection of Claims 2-9, 16, 18-22, 26, 28, 30-35, 37-45, and 53, withdraw the rejections, and allow Claims 2-9, 16, 18-22, 26, 28, 30-35, 37-45, and 53.

Examiner Interview

Applicants would like to thank the examiner for the interview conducted Friday, February 2, 2007 during which a number of pending claims and certain cited prior art was discussed.

NEWLY PRESENTED CLAIMS

Applicants added Claims 54-60 above. Applicants represent that the addition of claims 54-60 adds no new subject matter to the present application. Further, Applicants represent that the newly presented claims are supported by the instant specification and the provisional patent application to which is claims priority and all associated Figures. Applicants respectfully request that the examiner consider the newly presented claims and allow Claims 54-60.

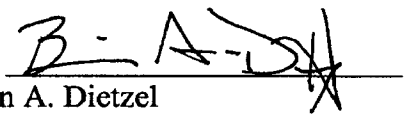
CONCLUSION

In light of the remarks set forth above, Applicants believe that they are entitled to a letters patent in the present matter. Applicants respectfully solicit the Examiner to expedite prosecution of this patent application to issuance. Should the Examiner have any questions or feel that further prosecution of this matter may be expedited through an interview, the Examiner is encouraged to telephone the undersigned.

The Commissioner is authorized to charge any additional fees which may be required, including petition fees and extension of time fees, to Deposit Account No. 23-2415 (Docket No. 26625-703).

Respectfully submitted,

Date: February 5, 2007

By: 
Brian A. Dietzel
Registration No. 44,656

WILSON SONSINI GOODRICH & ROSATI
650 Page Mill Road
Palo Alto, CA 94304-1050
(512) 338-5423
Client No. 021971